PASSED

Lab 6 group

Niclas Lovsjö, Maxime Bonneau

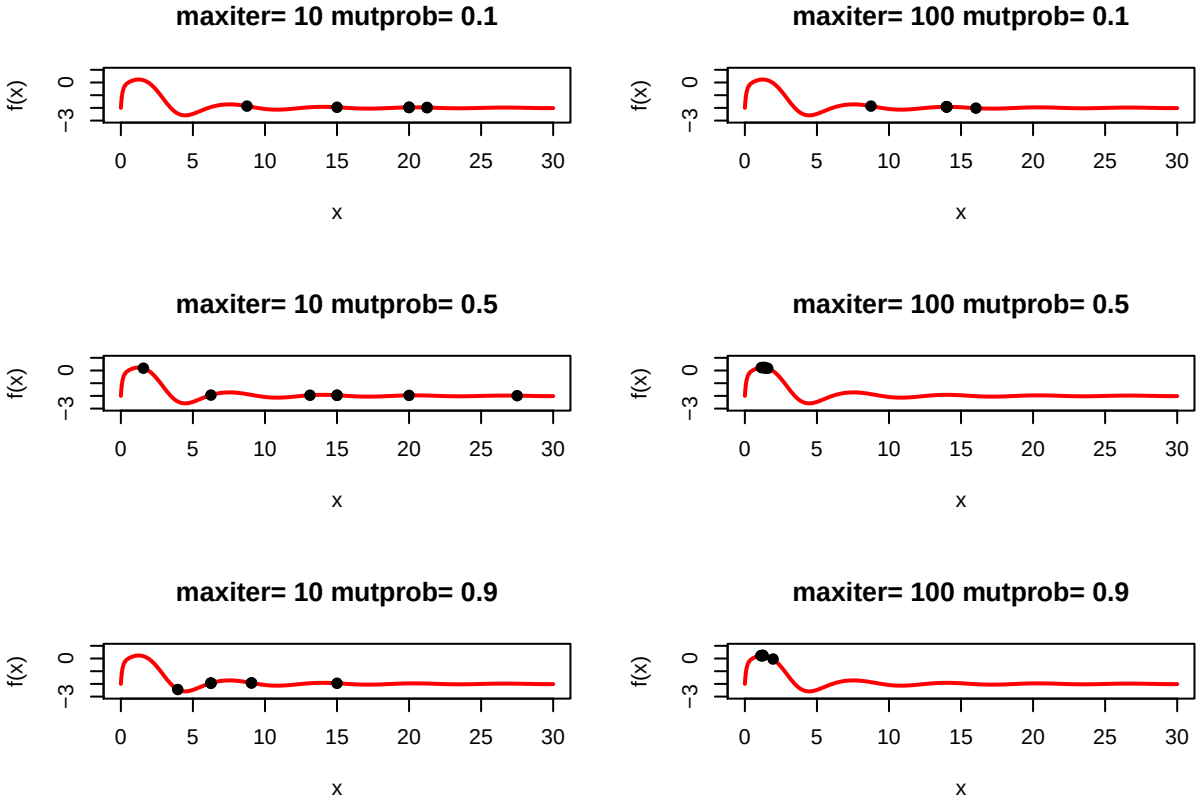
16 mars 2016

Assignment 1

We follow the lab-instructions for the first 4 questions, and then answer the questions in 1.5.

1.5

Run your code with different combinations of maxiter=10, 100 and mutprob=0.1, 0.5,0.9. Observe the initial population and final population. Conclusions?

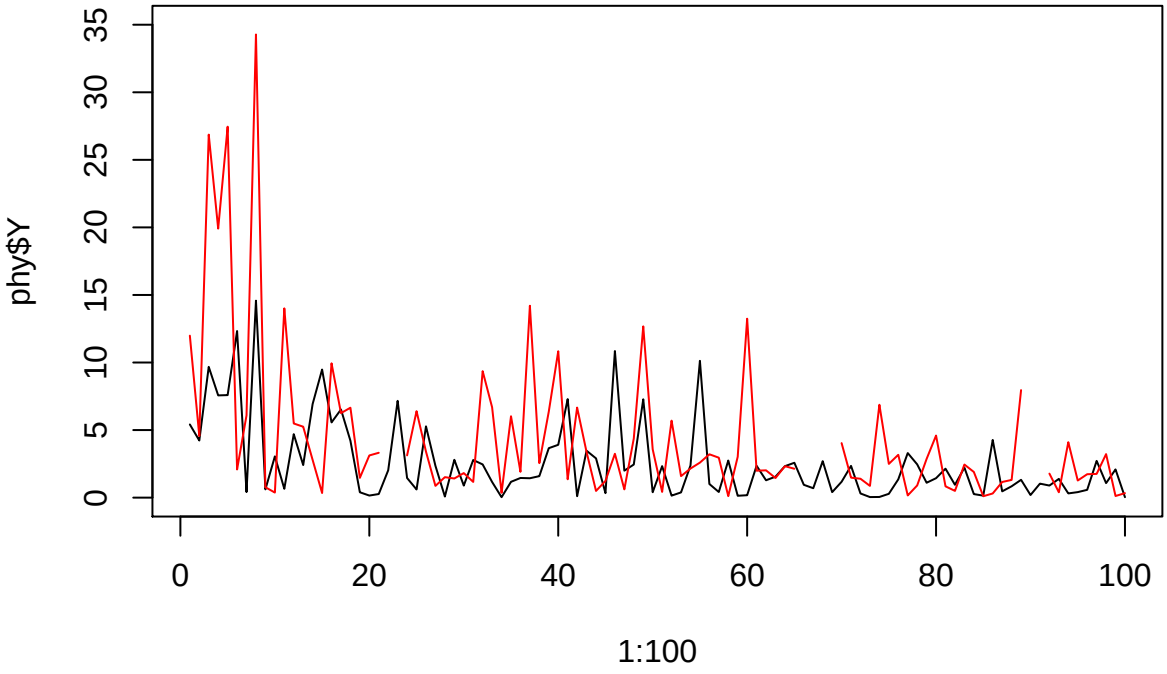


These plots show that while using maxiter= 10 the algorithm does not find the maximum, neither for maxiter= 100 with mutprob= 0.1. This is due to that there is a low probability of mutation. Mutation moves the point away from the current position, which then takes it away from local maximums. If this doesnt happen very often, the points will stay in their neighbourhoods. Furthermore we see that the last two combinations of maxiter= 100 all points is in a neighbourhood of the maximum point. However it can happen that not all points is in this neighbourhood for the last combinations either, but in general it seems that the very last (down right plot) is concentrated around the max point than the next to last (mid right). So as a conclusion, more iterations and higher chance of mutation gives the best result, as expected.

Assignment 2

1.

Under is the graph of Y (in black) and Z (in red) versus X. Y and Z do not seem to be related. The mean decreases when X increases for both Y and Z. It looks like Y and Z have the same kind of distribution.

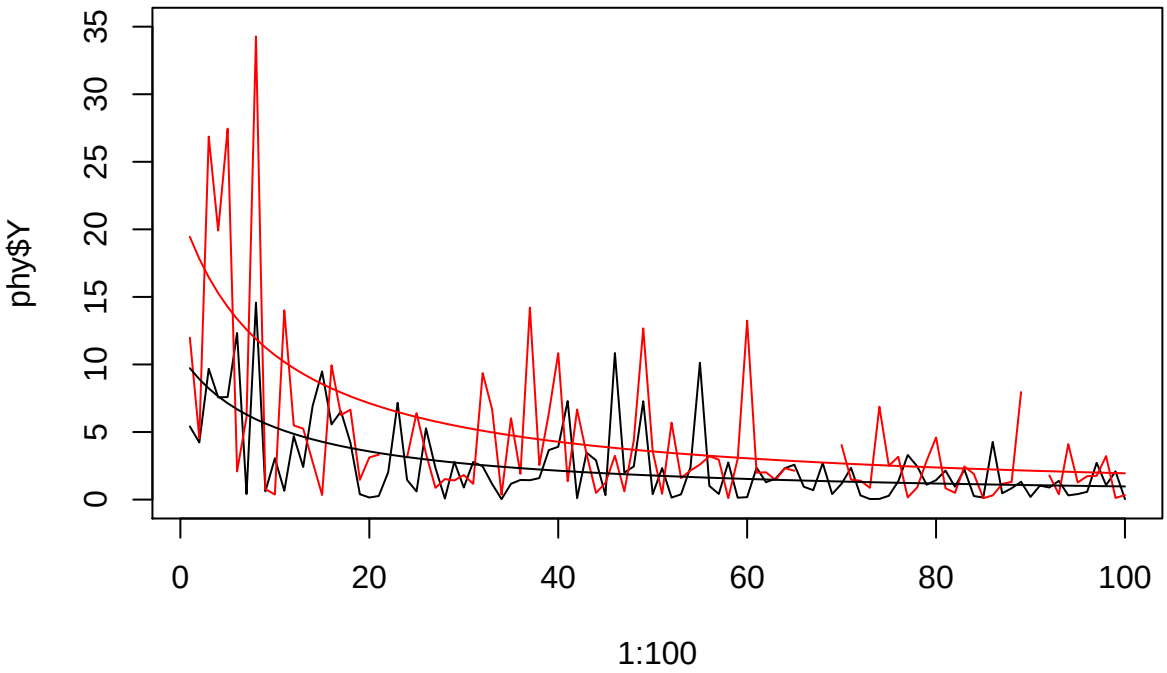


3.

```
## 14.26782 -414.9261
## 10.83853 -405.4854
## 10.70136 -405.3667
## 10.69587 -405.3625
## 10.69566 -405.3624
```

After computing the algorithm with a $\lambda_0 = 100$, we find an optimal λ of 10.6956555, after five iterations.

4.



When plotting EY and EZ on the same graph, we can see that the computed value of λ allows to follow globally the distribution of respectively Y and Z. So the result looks to be reasonable.

Contribution:

We did our labs separately, and then met and discussed them. Then we merged them, so that the first part comes from Niclas and the second from Maxime.

Code:

```
library(knitr)
opts_chunk$set(echo=FALSE)
setwd("/Users/niclaslovsjo/Library/Mobile Documents/com~apple~CloudDocs/Kurser/Comp stat/t6")
#1.1
f<-function(x){x^2/exp(x)-2*exp((-9*sin(x))/(x^2+x+1))}
#1.2
crossover<-function(x,y){(x+y)/2}
#1.3
mutate<-function(x){x^2%%30}
#1.4
main.fun<-function(maxiter,mutprob){
  set.seed(12345)
  x<-seq(0,30,0.01)
  #a
  plot(x,f(x),type="l",col="red",lwd=2,main=paste("maxiter=",maxiter,"mutprob=",mutprob),ylim=c(-3,1))
  #max a little over 0
  #b
  init.pop<-seq(0,30,5)
  #c
  values<-f(init.pop)
  #d
  iter<-0
  current.max<-c()
  current.max.index<-c()
  while (iter<maxiter) {
    iter<-iter+1
    #i
    parents<-sample(init.pop,2)
    #ii
    index.victim<-order(f(init.pop))[1]
    victim<-init.pop[index.victim]
    #order gives the indices as vector in order of magnitude for the function.
    #pick out smallest and use as selector in init.pop.
    #i.e. this chooses the smallest point and swaps for the new.kid, which be
    #mutated.

    #iii
    new.kid<-crossover(parents[1],parents[2])
    if(runif(1)<mutprob){
      new.kid<-mutate(new.kid)
    }
    init.pop[index.victim]<-new.kid
    values[index.victim]<-f(new.kid)
    current.max[iter]<-max(values)
    current.max.index[iter]<-init.pop[which.max(values)]
  }
  points(init.pop,f(init.pop),pch=19)
  return(init.pop)
}
par(mfrow=c(3,2))
m1<-main.fun(10,0.1)
m2<-main.fun(100,0.1)
```

```r
m3<-main.fun(10,0.5)
m4<-main.fun(100,0.5)
m5<-main.fun(10,0.9)
m6<-main.fun(100,0.9)

phy <- read.csv("physical.csv")
plot(x = 1:100, y = phy$Y, type = "l", ylim = c(0,35))
lines(phy$Z, type = "l", col = "red")
em.exp <- function(Y,Z){
  Zobs <- Z[!is.na(Z)]
  Zmiss <- Z[is.na(Z)]
  X <- phy$X
  Xobs <- phy$X[!is.na(Z)]
  n <- length(c(Zobs, Zmiss))
  r <- length(Zobs)
  # Initial values
  lambdat <- 100
  # Define log-likelihood function
  ll <- function(y, z, lambda){
    2*sum(log(X)-log(sqrt(2)*lambda))-sum(X/lambda*y)-sum(Xobs/(2*lambda)*z)
    }
  # Compute the log-likelihood for the initial values
  lltm1 <- ll(Y, Zobs, lambdat)
  repeat{
    # M-step
    lambdat1 <- sum(X*Y)/(2*n)+sum(Xobs*Zobs)/(4*n)+(n-r)*lambdat/(2*n)
    # difference in lambdas
    diff <- abs(lambdat-lambdat1)
    # Update parameter values
    lambdat <- lambdat1
    # Compute log-likelihood using current estimates
    llt <- ll(Y, Zobs, lambdat)
    # Print current parameter values and likelihood
    cat(lambdat, llt, "\n")
    # Stop if converged
    if ( diff < 0.001) break
    lltm1 <- llt
  }
  return(data.frame("Optimal lambda" = lambdat1, "value of log-likelihood" = lltm1))
}

em <- em.exp(phy$Y, phy$Z)
EY <- em$Optimal.lambda/phy$X
EZ <- 2*em$Optimal.lambda/phy$X
plot(x = 1:100, y = phy$Y, type = "l", ylim = c(0,35))
lines(phy$Z, type = "l", col = "red")
lines(EY)
lines(EZ, col = "red")
## NA
```